International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249 – 8958 (Online), Volume-9 Issue-5, June 2020

Comparative Study of Software Defect Prediction and Analysis the Class using Machine Learning Method

V. Ruckmani, S. Prakasam

Abstract: ***An automatic mode that increases sample stability is checked to verify the software design. Predict software flaws are the main focus of the engineering department. Computational software engineering is one of the active study areas of a software flaw. Depending on the metric, software quality and the efficient allocation of volume resources can easily improve defect quality, thus reducing costs. Many data mining and datasets can be used to store defect prediction software. Machine learning software defect prediction technology is an important branch of the computer. Therefore, in this method is to develop the defect prediction obtained by the design of selected class function metrics to create an effective error finding model. Various models have been proposed to reflect the changing changes in the software product's defect prediction index. These models also validate the data of the corresponding software module. The software defect analysis uses various software products for performance metrics to predict. It helps to find a different relationship between software volume and error size. Object classes are the user interface components in interactive applications. The control of the function property value assigned to the parsing code. The machine learning logic to detect errors due to defects. Advanced defect prediction models use different methods of performance class and function to evaluate. It provides a valid defect prediction for the defect identification code. This information is implemented in application software to improve predictive error classes and merit function code.***

Keywords: Computational software engineering, identification, application software

I. INTRODUCTION

Software research has always been an important topic in software engineering diagnostics. To look at the flaws in the software component, the current flaw in the job speculation software system is lies. The number of classification defects is usually divided into two categories, the defect is easier than the defects [1-2]. Criticism of the software defect prediction model [4,6] is the potential delivery quality and repair, allowing many companies to estimate the number of failures in a software system. Any software that can best be built with the help of many different measurements and geometric models. We offer serious criticism of these literary and state-of-the-art genres. The most widely used indicator for predicting the difficulty of prototyping for detecting regular size and errors. Other, "standard" or multidimensional methods in the development process, based on test data. Research has many serious theoretical and practical problems.

Revised Manuscript Received on June 30, 2020.
* Correspondence Author
V. Ruckmani*, Assistant Professor, Voorhees College Vellore, Anna Salai, Kosapet, Vellore, Tamil Nadu, India. E-mail: annaruckmani555@yahoo.com
S. Prakasam, Associate Professor, Scsvmv University, Enathur, Kanchipuram, Tamil Nadu, India.

So far, the model is weak and there are no known flaws or errors [10], so the relationship between them is not possible. Basic statistics and data quality models found in underrated notes.

A. Software Defect Prediction

Software engineering is one of the most active areas of software innovation. We can create software project data, Project Defect Detection (WPDP) in a single project under Defect Detection. The authors believe that cross-project defect detection (CPDP) is a new program and that other projects use damage data prediction model. Recent studies show CPDP potential [20]. However, CPDP Indicators must be of the same block, in any case, to be uniform objects between index sets. Therefore, the versatility of current technology, horizontal CPDP project applications and indicators [7] may be more appropriate.

Most of the existing methods are additional project defect detection (WPDP), so the serial forecasting model focuses on the latest data to detect failure trends in new software modules in the same project. However, researchers have found that other programs, whether they support statistical data, have to rely on their knowledge goals, and in cases where they have to rely on sufficient historical data.

B. Software Quality Assurance (SQA)

It is a deliberate, methodical approach that provides a software product with sufficient confidence that it can meet the requirements of the software development process. The SQA software development process includes developed software that includes quality and value assurance methods, tools, technologies, and technologies [23]. This ensures the integrity and reliability of commonly implemented software. SQA uses tools and quality control processes, including well-defined standard procedures. The importance of software management is software quality insurance. At various angles, software quality is not dedicated to improving quality research and related activities, and many decisions and procedures have been made for decades [24]. Research in this area suggests that an early feature of software can be used to establish quality assessment models to evaluate quality in the project.

C. Software Reusability

The software is invisible, but it must be a successful system. Software reliability should be measured and evaluated. Software reliability must be established in all software industries.

They are normalized by software templates so that they are not artificially modified or upgraded by programmers [26-28]. The main objective of measuring software reliability is usually released during the design process. Reliability is usually a work process within a given time period, in productivity under certain conditions. Software reliability may indicate barriers to running this particular software. It is defined as a software product with a degree of software reliability. It is impossible to specify a software solution, how to measure the reliability of the solution, and the associated software product. Due to the lack of the same amount of software products, it is impossible to measure the reliability of the characters used in the study [30].

D. Software Defect Prediction (SDP) Various Method

Another method has been used to evaluate defects. Machines need to improve their efficiency to look at the additional benefits of their performance and learning-based methods, but there is still no time available for data [22].

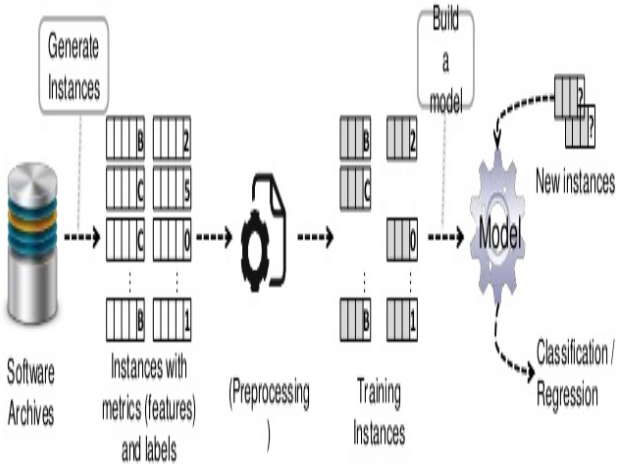


Figure 1 Defect analysis process

(https://www.slideshare.net/hunkim/20140703-survey-onsoftwaredefectprediction4th)

The software learning model is not difficult to obtain when a software project has proved important, despite the prediction of software deficiency, and is shown in Figure 1., they are predicting the importance of using SDP ML technology only because of the shortcomings and the lack of experience due to the fact that professionals and application practitioners are clearly promoted. Although the complexity of the method depends on the different aspects of the technique used, the earlier paper began to focus on the SDP prediction accuracy [21].

II. RELATED WORK

In this article, the Golan Mausa Analyzer Error Code is used to demonstrate a software flaw prediction data collection tool [16], an experienced software engineer, who is a suite of learning datasets in real software development projects. The aim is to focus on software engineering. Factors that frequently deviate from building-related data sets do not reduce theoretical knowledge.. This problem is resolved by the data collection system of the data collection program, the employee who has resolved, the quality of the open community, and the creation of mechanisms. This article's gift tools, descriptions, data collection programs, open source error tracking, and source control library software for predicting bad data.

Retrieval Timing Analysis Embedded Software This paper [29] proposes a new method of reconstruction software standard timing analysis software. Its purpose is to create a software development environment to determine the time limit issues that have taken place in modern processor migration to solve this problem. Another contribution of this paper, such as branch guessing, inference and prediction algorithms, has a significant impact on the performance of the system, but is an important feature of very few functional models in relation to modern architecture.

Bayesian prediction fuzzy technology enables interactive fault analysis. Liyang [1] proposed a new barrier analysis and prioritization method. Both vague and Bayesian predictions have strengths. Fuzzy Fuzzy Rule-Based Bayesian Prediction (FuRBaR) is an error model based on the number of changes to estimate the fuzzy model. The study of simulated and increasingly complex reliability problems requires accurate failure data. Providing these vital statistics is helpful. Jaisang Nam et al. [3] A common understanding is that communication developer behavior affects the quality of a developer, such as the behavior of a developer, and the impact of developer behavior on predicting these widely used disadvantages. There is no such thing. Therefore, we recommend that you use the Micro Interaction Indicator (MIM), which developer contact information you use. Interactive meetings with developers to edit tasks such as viewing, capturing, and storing Mylyn information. Shams Hu et al. The real flaw is that programs (i.e, the most important indicator selected) are those that are more predictable and more compact indicators or filtering methods than traditional packaging. The implementation of the proposed framework is also verified by statistical multiple quality controllers using Multivariate Exponentially Weighted Moving Averages (MEWMA) process masses. It is proposed that the combination of two containers can be achieved by computing the best way of computing composite computing to guide the metric selection process through the filtering advantages of the inherent properties of the filter. Qiao Yu et al. [7] Thus, the disadvantages introduced by the selective detection software (WPDP's appropriate item defect assumption) can improve conventional prediction capabilities. However, this cross section does not suffice to select CPDP research activity. This subset uses a selection of features to evaluate the CPDP's key features in its evaluation system. Xiao Asai Jing et al. [8] The internal gap between the two modes and the horizontal method provides an effective solution. A convenient every minute (if not done) mode of learning is introduced to solve this problem through role play. You can learn more powerful classification features from original indicators.

Subsidiary projects do not achieve equilibrium, and advanced violation (ISDA) methods have been proposed.

The project proposes the SSTCA + ISDA forecasting method using the semi-monitored written instrument analysis (SSTCA) method that generates targeted data delivery through project forecasting.

FEI Wu et al. [9] Source data tags can only be used if they are not appropriate. In this case, the Cross Project Defect Detection (CPDP) is called the Semiconductor Teacher Planned Cross Defect Detection (CSDP). Semi-monitored Defect Detection System (WSDP) Many recently developed programs still have room to predict performance improvements. As a unified and effective solution, we propose to introduce a learning kernel and a cost-conscious Atomic Semi-Sensitive Learning Dictionary (CKSDL) system under the supervision of half-teaching semi-teacher. There is a limited amount of data available in the kernel area and the coding data is coded with the CKSDL tag.

Martin Shepperd et al. [11], the value of software components as flaws is simple. So, in order to evaluate the various technological initiatives that are being run effectively, some research indicates that research has been done. However, as with any dominant technique, reliable defect prediction models are still examples of problem design. Attempting to understand that many conflicting emotions make decisions has a huge impact on performance. In all relevant, high-quality early-stage studies, Meta-analyzes were performed to determine whether the factors predictive of defects affect performance. This is a 42 week keynote study based on our report on the 600 boxes of empirical forecasts. The variance model utilizes generalized random effects analysis variables (classification, data collection, input metrics, and research team) to model the relative performance of the considered performance established by four methods and inverse engineering.

Ebene Bennin [13] introduced that MAHAKIL software is a new and effective set of methods for data mining that relies on chromosome genetics oversampling theory. In order to create a new phenomenon using this philosophy, MAHAKIL, as a parent, described the different roles that two different subgroups possess as different properties and distribute data from different levels.

Hai Tao et al. [15] proposed an EMR_SD to predict Multi Boost in an unbalanced software flaw data population based on the RIPPER classification. First, the original features are from the data using the principal component analysis (PCA) method, the most effective feature of this algorithm is the reduction of the evolution and elimination of unnecessary targets.

Zhongbin sun et al. [16] the lack of software makes it difficult to predict the unbalanced nature of the data. For this reason, many problems need to be addressed. However, these traditional methods require access to raw data, key learning delivery, variable costs, package building, and, if possible, sampling errors that provide important information. The first binary unbalanced conversion paper rendering equilibrium is a new method for generating multiple types of predictive data for specific types of encoded data and defects.

In the research by XIE and CAI et al. [20] No sample code is provided for neural networks, and the actual number of ASTs is an encrypted abstract syntax hierarchy. However, for most cross-scheme defect detection (CPDP) representative defect predictions, the semantic distance between ASTs can be an important way to effectively reduce the semantic distance from reality estimated training ability, and it can be converted to Tree Based Embedding (TBE), the new symbolic frame, which combines the problem of semantic distance embedding between the tree and the tree structure and converts it to a real vector.

Mingxia Liu et al. [22] Easy to create. Instead of classification level-only function selectivity, we have the new two-stage cost-oriented learning (TSCS) method, which is recommended for you to use SDP and payment information.

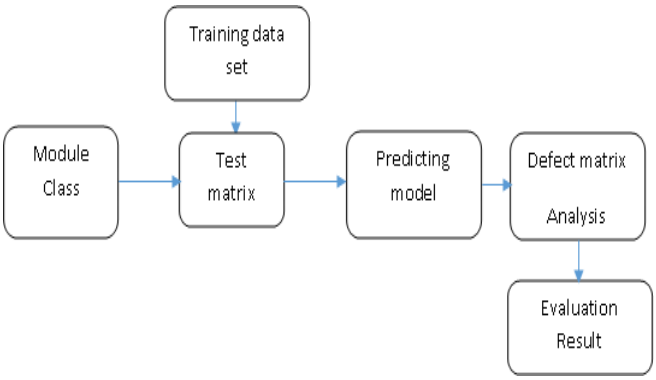
Reference	Objectives	Methodology	Parameter	Issues Solved	Unsolved & Demerits	Remarks
[3]	Micro Interaction Indicator (MIM). This balance is the way of developer contact information. Interactive developers, such as editing files and viewing event work sessions, are captured and stored with Mylyn information.	micro-interaction metrics (MIMs)	According to the assessment, MIM can improve the accuracy of predicting overall defects. And here the distribution of F values improve the mean.	According to the assessment, MIM can improve the accuracy of predicting overall defects. Give developers intuitive feedback that their behavior will be perceived as ineffective in the software development process.	MIM presents a variety of ITE-centric tools, including significant changes in risk of alerting promises before submission.	MIM promises submission pre-warrants that provide a variety of tools for the ITE hub, including a significant change in risk.

[5]	Two new composite software defect prediction models have been proposed to identify the key characteristic combinations and packaging techniques used in filters (of indexes). Different pricing methods, including SVM and ANN that combine the most appropriate (M), filtering methods to find the metrics.	Multivariate Exponentially Weighted Moving Average (MEWMA)	To solve the problem, look for greater predictive accuracy than traditional blanket and filter methods. Evaluate and predict subdivision metrics and features.	It takes a lot of measurements and can be slow and expensive to track. It also reduces the complexity of the selected metric and makes the predictive metric selection the same.	Current research is limited to procedural indicators.	It can also investigate the application of parallel computing to reduce computational complexity.
[7]	Use a subset of this subset of features to consider the validity of CPDP feature selection in ranking mode.	cross-project defect prediction (CPDP)	Selection and Features This method of cross-classizing a subset of cross-functional projects can improve predictable defect performance.	Selection and Features This method of cross-classizing a subset of cross-functional projects can improve predictable defect performance.	Selection and selection improve predictable impairment performance, and cross-classicalism may result in a subset of cross-functional projects in this way.	This may require selecting a representative subset of features or a set of reasonable percentages of selected features to improve CPDP performance in future studies.
[8]	Our goal is to provide an effective cross-project solution for the imbalance of both programs and classes.	Semi-supervised transfer component analysis (SSTCA) method, improved SDA (ISDA) method	ISDA-based solutions to work with other units of equilibrium methods in modern projects.	Significant impact of performance and predictive volatility cross-schemes is planned, and we are combining these two issues, which is an effective predictive frame.	In the case of multi-component cross-schemes that evaluate the effectiveness of our framework, variability in such complex types of cross-schemes is expected.	It enhances our learning system and our non-linear classification capabilities, including our sense of kernel-space attitude.
[9]	We present the semi-tracked dictionary learning technology and the cost-conscious atomic semi-tracked dictionary learning (CKSDL) method. You can use CKSDL defined tag data to obtain anonymous large amounts of data in the kernel area. CKSDL also considers the cost of classification when learning a dictionary.	cost-sensitive kernelized semi-supervised dictionary learning (CKSDL)	WSDP improves performance, much more easily than the associated SSDP, which delays data between projects, and, in a typical CSDP project, CKSDL provides.	Cross-project flaw can improve data performance. WSDP and CKSDL perform significantly better predictive performance than common SSDP scenarios.	Contraindications Data is included in many projects to test our approach to its use in general. So the harder function is to analyze code bugs.	Our approach to multiple, cross-project data can solve problems.
[11]	We get a lot of contradictions and manage the emotional effects in order to understand the factors that have a significant impact on the performance of our predictions.	random-effects ANOVA	To establish random effects analysis of the variance model, we are looking for four models that generate components regarding the relative contributions of the predictive model's performance (classifiers, datasets, input indicators, and study groups).	In this high-level research code to order, researchers must forecast (1) blindly analyze findings, (2) report improved protocols, and (iii) do more and more research teams to reduce our expertise issues.	The binary defect classification is expected to reflect this type of problem in predictive systems studied in other areas.	Reporting distortion issues is difficult to imagine with many branches of science.

[13]	With the results of these methods, most of them often lead to over-generalization (high false alarm rate) repetitive data instances (fewer different data).	MAHAKIL, Borderline-SMOTE, PROMISE repository	MAHAKIL improves predictive performance of all models and delivers more important predictive values than others oversampling techniques.	MAHAKIL improves the predictive performance of all models and delivers future deposit values that are more important than other oversampling techniques.	A cross-project or company forecast will have the same characteristics of domains as any preparatory programs in the historical data set that new training can use.	It will consider these projects in future studies to improve our approach. Finally, linear algebra MAHAKIL, can be used to explain the principle of functioning in intrinsic component analysis.
[15]	This classification uses the attributes classes to reduce the correlation rules for deviations and the variants, based on MultiBoost and aiming to reduce classification errors.	MultiBoost, Sampling method of adaptive synthetic sampling (ADASYN), and random sampling.	The NASA MDP general dataset is verified by experience with prediction models and similar methods are compared. Increases accuracy, p-number, AUC, and balance signs.	As far as metrics are concerned, the EMR_SD algorithm surpasses DNC, CEL, and other flaw prediction techniques that prove the usefulness of the algorithm.	In this article we will not implement all dataset methods by focusing on the data without considering the characteristics of each dataset.	Interest and Sampling Techniques Dimension Reduction All different datasets are used.
[16]	Many methods have been used to solve this problem because it is very difficult to predict the nature of software flaw data class imbalance.	Random Correction Code-Based Method, Bagging, and Boosting,	This time the compositions were low, high cost sensory oversampling, and the learning system was improved, the RUS RUS packing system was equally effective.	These shows are called methodology and coding, on average, more than the traditional method.	It suffers from domain imbalance issues rather than predicting flaws in many domains of software.	We need to improve the performance of more basic classification algorithms using our method.
[20]	We have called Transitional Hybrid Functional Learning (TBCNN-THFL) convolutional neural networks embedded in the pyramidal system to evaluate the ability of CPDP tasks and verification coding to perform convolutional neural networks. TBCNN-THFL Feed The TBE method for transferring data between different projects is used to identify general features.	TBCNN-THFL, convolutional neural network	It proves that the reference model of the CPDP work with sufficient performance number is better than the one developed by the open source project on TBCNN-THFL 729.	This improves the CBOW model and introduces the CPDP industry. To verify the performance of the improved CBOW model, TBE also proposes a framework for predicting defects.	It traditional manual features in the feature extraction phase.	This feature is a traditional manual feature during the extraction phase. The test was carried out on 72 CPDP tasks that resulted in nine promising projects.

III. IMPLEMENTATION METHOD DISCUSSION

In a previous related work, several methods were discussed to analyze software defects. Applications and methods Evaluate code applicable functions and other method classes to provide better performance. Any measure of software or software process product performance measured in relation to various elements and dimensions used to measure the dimensions of a software product or process. The tool is adjusted to predict future progress and early obstacle detection and planning.



`Figure 2 Software Code analysis model

The measures are then based on the characteristics of the proposed class matrices for evaluating software products, functions, and kernel status, and to determine the products or services offered and customer satisfaction. Research in its implementation technology shows the basic idea of software measurement to improve prediction accuracy and F-action processing. The above Figure 2 software flaw analysis and evaluation model is shown in the overall diagram.

IV. CONCLUSION

This discussion examines the drawbacks of using data mining software modules and predicting software metrics. Drawing upon a thorough overview of existing classification modules, which are related to the assumption of impairment in the literature due to the shortcomings that many empirical test results have, they are the general conclusion that this article should include the indicators we choose. This method selects the best functions based on the objective function. A new method of preprocessing defects based on software metrics. Monitored classification algorithms are used to predict various shortcomings.

REFERENCE

1. Song, Q., Jia, Z., Shepperd, M., Ying, S., & Liu, J. (2011). A General Software Defect-Proneness Prediction Framework. IEEE Transactions on Software Engineering, 37(3), 356–370.
2. Zhang, F., Hassan, A. E., McIntosh, S., & Zou, Y. (2017). The Use of Summation to Aggregate Software Metrics Hinders the Performance of Defect Prediction Models. IEEE Transactions on Software Engineering, 43(5), 476–491.
3. Lee, T., Nam, J., Han, D., Kim, S., & Peter In, H. (2016). Developer Micro Interaction Metrics for Software Defect Prediction. IEEE Transactions on Software Engineering, 42(11), 1015–1035.
4. Tantithamthavorn, C., McIntosh, S., Hassan, A. E., & Matsumoto, K. (2016). Comments on "Researcher Bias: The Use of Machine Learning in Software Defect Prediction." IEEE Transactions on Software Engineering, 42(11), 1092–1094.
5. Huda, S., Alyahya, S., Mohsin Ali, M., Ahmad, S., Abawajy, J., Al-Dossari, H., & Yearwood, J. (2018). A Framework for Software Defect Prediction and Metric Selection. IEEE Access, 6, 2844–2858.
6. Shepperd, M., Hall, T., & Bowes, D. (2017). Authors' Reply to "Comments on 'Researcher Bias: The Use of Machine Learning in Software Defect Prediction.'" IEEE Transactions on Software Engineering, 1–1.
7. Yu, Q., Qian, J., Jiang, S., Wu, Z., & Zhang, G. (2019). An Empirical Study on the Effectiveness of Feature Selection for Cross-Project Defect Prediction. IEEE Access, 1–1.
8. Jing, X.-Y., Wu, F., Dong, X., & Xu, B. (2017). An Improved SDA Based Defect Prediction Framework for Both Within-Project and Cross-Project Class-Imbalance Problems. IEEE Transactions on Software Engineering, 43(4), 321–339.
9. Wu, F., Jing, X.-Y., Sun, Y., Sun, J., Huang, L., Cui, F., & Sun, Y. (2018). Cross-Project and Within-Project Semisupervised Software Defect Prediction: A Unified Approach. IEEE Transactions on Reliability, 67(2), 581–597.
10. Yu, T., Wen, W., Han, X., & Hayes, J. (2018). ConPredictor: Concurrency Defect Prediction in Real-World Applications. IEEE Transactions on Software Engineering, 1–1.
11. Shepperd, M., Bowes, D., & Hall, T. (2014). Researcher Bias: The Use of Machine Learning in Software Defect Prediction. IEEE Transactions on Software Engineering, 40(6), 603–616.
12. Felix, E. A., & Lee, S. P. (2017). Integrated Approach to Software Defect Prediction. IEEE Access, 5, 21524–21547.
13. Bennin, K. E., Keung, J., Phannachitta, P., Monden, A., & Mensah, S. (2018). MAHAKIL: Diversity Based Oversampling Approach to Alleviate the Class Imbalance Issue in Software Defect Prediction. IEEE Transactions on Software Engineering, 44(6), 534–550.
14. Liang, H., Yu, Y., Jiang, L., & Xie, Z. (2019). Seal: A Semantic LSTM Model for Software Defect Prediction. IEEE Access, 7, 83812–83824.
15. He, H., Zhang, X., Wang, Q., Ren, J., Liu, J., Zhao, X., & Cheng, Y. (2019). Ensemble MultiBoost based on the RIPPER classifier for the prediction of imbalanced software defect data. IEEE Access, 1–1.
16. Sun, Z., Song, Q., & Zhu, X. (2012). Using Coding-Based Ensemble Learning to Improve Software Defect Prediction. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 42(6), 1806–1817.
17. Yang, X., Tang, K., & Yao, X. (2015). A Learning-to-Rank Approach to Software Defect Prediction. IEEE Transactions on Reliability, 64(1), 234–246.
18. Huda, S., Liu, K., Abdelrazek, M., Ibrahim, A., Alyahya, S., Al-Dossari, H., & Ahmad, S. (2018). An Ensemble Oversampling Model for Class Imbalance Problem in Software Defect Prediction. IEEE Access, 6, 24184–24195.
19. Zhang, Z.-W., Jing, X.-Y., & Wu, F. (2018). Low-rank representation for semi-supervised software defect prediction. IET
20. Cai, Z., Lu, L., & Qiu, S. (2019). An Abstract Syntax Tree Encoding Method for Cross-Project Defect Prediction. IEEE Access, 7, 170844–170853.
21. Song, Q., Guo, Y., & Shepperd, M. (2018). A Comprehensive Investigation of the Role of Imbalanced Learning for Software Defect Prediction. IEEE Transactions on Software Engineering, 1–1.
22. Liu, M., Miao, L., & Zhang, D. (2014). Two-Stage Cost-Sensitive Learning for Software Defect Prediction. IEEE Transactions on Reliability, 63(2), 676–686.
23. Doherty, B., Jelfs, A., Dasgupta, A., & Holden, P. (2016). Defect Analysis in Large Scale Agile Development: Quality in the Agile Factory Model. 2016 Joint Conference of the International Workshop on Software Measurement and the International Conference on Software Process and Product Measurement (IWSM-MENSURA).
24. Venkatasubramanyam, R. D., Gupta, S., & Uppili, U. (2015). Assessing the Effectiveness of Static Analysis through Defect Correlation Analysis. 2015 IEEE 10th International Conference on Global Software Engineering.
25. Yusop, N. S. M., Schneider, J.-G., Grundy, J., & Vasa, R. (2017). Analysis of the Textual Content of Mined Open Source Usability Defect Reports. 2017 24th Asia-Pacific Software Engineering Conference (APSEC).
26. Imparato, A., Maietta, R. R., Scala, S., & Vacca, V. (2017). A Comparative Study of Static Analysis Tools for AUTOSAR Automotive Software Components Development. 2017 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW).
27. Chi, J., Honda, K., Washizaki, H., Fukazawa, Y., Munakata, K., Morita, S., Yamamoto, R. (2017). Defect Analysis and Prediction by Applying the Multistage Software Reliability Growth Model. 2017 8th International Workshop on Empirical Software Engineering in Practice (IWESEP).
28. Christa, S., & Suma, V. (2016). An analysis of the significance of ticket analytics and defect analysis from a software quality perspective. 2016 International Conference on Inventive Computation Technologies (ICICT).
29. Yamashita, A. (2015). Experiences from performing software quality evaluations via combining benchmark-based metrics analysis, software visualization, and expert assessment. 2015 IEEE International Conference on Software Maintenance and Evolution (ICSME).
30. Verma, D. K., & Kumar, S. (2015). The empirical study of defects dependency on software metrics using a clustering approach. 2015 IEEE UP Section Conference on Electrical Computer and Electronics (UPCON).